Content-based image retrieval with compact deep convolutional features

Alzu'bi, Ahmad; Amira, Abbes; Ramzan, Naeem

Published in:
Neurocomputing

DOI:
10.1016/j.neucom.2017.03.072

Published: 02/08/2017

Document Version
Peer reviewed version

Content-Based Image Retrieval with Compact Deep Convolutional Features

Ahmad Alzu'bi [a], Abbes Amira [a,b], Naeem Ramzan [a]

[a] School of Engineering and Computing
University of the West of Scotland, Paisley, PA1 2BE, UK

[b] KINDI Center for Computing Research, Qatar University, Qatar

(ahmad.alzubi; abbes.amira; naeem.ramzan)@uws.ac.uk

Abstract. Convolutional neural networks (CNNs) with deep learning have recently achieved a remarkable success with a superior performance in computer vision applications. Most of CNN-based methods extract image features at the last layer using a single CNN architecture with orderless quantization approaches, which limits the utilization of intermediate convolutional layers for identifying image local patterns. As one of the first works in the context of content-based image retrieval (CBIR), this paper proposes a new bilinear CNN-based architecture using two parallel CNNs as feature extractors. The activations of convolutional layers are directly used to extract the image features at various image locations and scales. The network architecture is initialized by deep CNNs sufficiently pre-trained on large generic image dataset then fine-tuned for the CBIR task. Additionally, an efficient bilinear root pooling is proposed and applied to the low-dimensional pooling layer to reduce the dimension of image features to compact but high discriminative image descriptors. Finally, an end-to-end training with backpropagation is performed to fine-tune the final architecture and to learn its parameters for the image retrieval task. The experimental results achieved on three standard benchmarking image datasets demonstrate the outstanding performance of the proposed architecture at extracting and learning complex features for the CBIR task without prior knowledge about the semantic meta-data of images. For instance, using a very compact image vector of 16-length, we achieve retrieval accuracy 95.7% (*m*AP) on Oxford5K and 88.6% on Oxford105K; which outperforms the best results reported by state-of-the-art approaches. Additionally, a noticeable reduction is attained in the required extraction time for image features and the memory size required for storage.

Keywords: CBIR**;** Deep learning; Convolutional neural networks; Bilinear compact pooling; Similarity matching

1. INTRODUCTION

In the domain of content-based image retrieval (CBIR), the retrieval accuracy is essentially based on the discrimination quality of the visual features extracted from images or small patches. Image contents (objects or scenes) may include different deformations and variations, e.g. illumination, scaling, noise, viewpoint, etc, which makes retrieving similar images one of the challenging vision tasks. The typical CBIR approaches consist of three essential steps applied on images: detection of interest points, formulation of image vector, and similarity/dissimilarity matching.

In order to extract representative image features, the most existing CBIR approaches use some hand-crafted low-level features, e.g. scale-invariant features transform (SIFT) [1] and speed-up robust features (SURF) [2] descriptors. Such features are usually encoded by general orderless quantization methods such as vector of locally aggregated descriptors (VLAD) [3]. The resulting image representations have shown a high capability on preserving the local patterns of image contents by capturing local characteristics of image objects, e.g. edges and corners. Therefore, they are suitable for the image retrieval task and widely used for matching local patterns of objects. However, convolutional neural networks (CNNs) have recently demonstrated a superior performance over hand-crafted features on image classification [4-6]. Adopting a deep learning procedure on multiple layers of convolutional filters makes CNNs able to subjectively learn even complex representations for many vision and recognition tasks.

Many recent works [5,7,8] demonstrate that the CNN-based generic features adequately trained on sufficient and diverse image datasets, e.g. ImageNet [9], can be successfully applied to other visual recognition tasks. Additionally, performing a proper fine-tuning on CNNs using domain-specific training data can achieve a noticeable performance in common vision tasks [5,10]; including object localization and instance image retrieval. Despite the promising results achieved by CNNs so far, there is no exact understanding or common agreement on how these deep learning architectures work; especially at the intermediate hidden layers. Several successful approaches [11-14] have applied CNNs to extract generic features for image retrieval tasks and obtained promising results. They mainly utilize the power of local features to generate a generic image representation based on some pre-trained CNNs. Nevertheless, many open questions and challenges need more investigation. Foremost, the effectiveness of fine-tuning the CNN models pretrained for specific task, e.g. image classification, on transfer learning to the CBIR task. Secondly, the discrimination quality of image features

directly extracted from the convolutional layers compared to the features quantized using the traditional generic approaches such as VLAD. Thirdly, the ability of reducing the unfavourable high-dimensional image representations generated by the most of existing CNN-based architectures. Finally, a proper investigation is required on how efficient connections can be made between several CBIR aspects; including query handling, similarity/dissimilarity matching, and retrieval performance in term of search time and memory usage. All of these challenges motivated us to develop and utilize a different deep CNN architecture in order to address the problems associated with features quantization, model fine-tuning, high-dimensionality, and system performance affected by the training procedure and features lengths.

Accordingly, the main aim of this paper is to propose a new CNN-based learning model in the context of CBIR. The proposed architecture is inspired by the bilinear models proposed by Tenenbaum and Freeman [15] to model the separation between the "content" and "style" factors of perceptual systems and by the promising results obtained using bilinear CNNs applied to fine-grained categorization [16]. Specifically, two parallel CNNs are adopted to directly extract image features from the activations of convolutional layers using only the visual contents and without prior knowledge about the semantic meta-data of images, i.e. no tags, annotations, or captions have been used. Image representations are generated by accumulating the extracted features at image locations and scales in order to model local feature correlations. The proposed architecture is initialized by pre-trained deep CNN models that adequately fine-tuned in unsupervised manner to learn the parameters for CBIR tasks using several standard retrieval datasets. Moreover, an efficient compact root pooling layer is also proposed based on the compact bilinear pooling recommended by Gao et al. [19], which demonstrates a noticeable improvement in the retrieval accuracy. Most critically, the resulting final image vectors are very compact so they reduce the time needed to extract them and reduce the memory size required to index the images and architecture with its parameters. Finally, the discriminatory capability of the image descriptors obtained by the proposed model is examined on different CBIR tasks, e.g. general, object-focused, landmarks image retrieval, and large-scale image retrieval.

The remaining part of this paper is organized as follows: Section 2 review the related works in literature; Section 3 presents the proposed compact bilinear architecture along with complete retrieval framework; Section 4 demonstrates and discusses the experiments carried out on several standard image retrieval datasets; and Section 5 concludes this work.

2. RELATED WORK

The most commonly CNNs architectures used in the CBIR are initially trained for classification tasks, where the representations extracted from the higher layers of CNN networks are usually used to capture semantic features for the category-level classification. Transfer learning of generic CNN features, trained on very large classification-based image datasets, to be used for image retrieval has shown a noticeable performance by several works. Wan et al. [11] applied many existing deep learning methods for learning feature representation from images and their similarity measures with application to CBIR tasks, e.g. object and landmark image retrieval. They concluded that a direct use of features extracted from deep CNN can further boost the retrieval performance and outperform the hand-crafted features. However, authors used very large vocabulary sizes to construct image vectors using the bag-of-word (BOW) encoding approach, which degrades the retrieval performance in terms of training time and memory storage.

Most recently, the use of VLAD and fisher vectors (FV) quantization methods has been increased due to their effectiveness in image retrieval applications. Several works [30-32] have improved these approaches in order to aggregate the extracted features from the image into a generic representation. Accordingly, the VLAD and its variants have also been recently applied in the context of CNN-based image retrieval; especially at the higher layers, i.e. the output layers. Gong et al. [12] extracted CNN feature activations of local image patches at multiple scale levels then they performed the VLAD orderless quantization to pool these features at each level separately and generate a generic image representation, referred as multiple orderless pooling (MOP-CNN). Ng et al. [13] adopted a similar retrieval scheme by encoding the extracted features from deep CNNs at multiple scales and layers into a single VLAD vector. Yandex and Lempitsky [14] also aggregated local CNN features into a general compact image vector based on sum pooling, which showed a better performance than other shallow-based and CNN-based models. Razavian et al. [17] utilized a spatial search based on small and medium image representations extracted from convolutional networks (ConvNets) for visual image retrieval. Paulin et al. [18] used patch-level descriptors by adapting a convolutional kernel network, i.e. batch-CKN, to perform an unsupervised learning of explicit feature embedding for both batch and image retrieval.

However, the majority of recent CNN-based approaches used in the CBIR domain have concatenated image features extracted at multiple scales then encoded by VLAD or BOW, and thus they have not considered the direct extraction of image features from the lower CNN layers for CBIR tasks. However, convolutional layers can be much more specialized and efficient than fully connected layers. Additionally, there are no assumptions about image features and patterns extracted from fully connected layers or by general aggregation methods since

they act as a general-purpose connection patterns. Moreover, they have an expensive performance cost in terms of memory and computations. The existing CNN-based CBIR approaches have also focused on the number and type of CNN layers used for feature extraction with using conventional pooling, e.g. max and sum pooling, and dimension reduction approaches, e.g. principal component analysis (PCA). Accordingly, more investigation is required on the effectiveness of image representations extracted and pooled directly from convolutional layers and then reduced to very compact lengths.

Unlike all existing CNNs and other deep models, we introduce a new deep bilinear CNN architecture in the context of visual CBIR. The proposed model is inspired by the successful application of bilinear models, which was first introduced by Tenenbaum and Freeman [15], in the context of image classification. Lin et al. [16] applied a bilinear CNN-based architecture for fine-grained image categorization using two feature extractors. The resulting features are multiplied by outer product at each image location and pooled to form the final image descriptor. However, our proposed architecture and aims are different from their bilinear model in many directions. Firstly, our model is based on unsupervised feature learning extracted only from the visual image content, and therefore it does not depend on class labels, annotations, or bounding boxes. Secondly, the dimension of the extracted features is extremely reduced into a low-dimensional image representation using a modified pooling scheme (Section 3) of the compact bilinear pooling recommended by Gao et al. [19], which is crucial in terms of computation and memory usage. To the best of our knowledge, this is the first work employs the bilinear CNN-based architectures in the context of CBIR, which also performs transfer learning on several tasks, e.g. object, landmark retrieval, and large-scale image search. The experimental results achieved under different scenarios emphasize the efficiency of our deep learning model in the CBIR domain, which can further boost the retrieval performance in terms of accuracy, extraction and search speed, and memory usage.

3. The Architecture

3.1. The Framework of Retrieval and Deep Learning

Our approach consists of three main steps: 1) Initialize the architecture by deep CNN networks pre-trained on millions of images; 2) fine-tune the bilinear CNN architecture on image retrieval datasets, i.e. transfer learning; and 3) extract features of query and dataset images. As shown in Figure 1, the CNN architecture is based on two variants of recent neural networks [20]: imagenet-vgg-m (VGG-m) and imagenet-vgg-verydeep-16 (VGG-16), and both are pre-trained on ImageNet [9]. These CNNs consists of convolutional layers, pooling layers, and fully connected layers. Both networks take images of size 224×224 pixels as input. To simplify the experiments, the fully connected layers are discarded from the fine-tuned bilinear CNN architecture, so that all image features are directly extracted from the activations of convolutional feature maps.

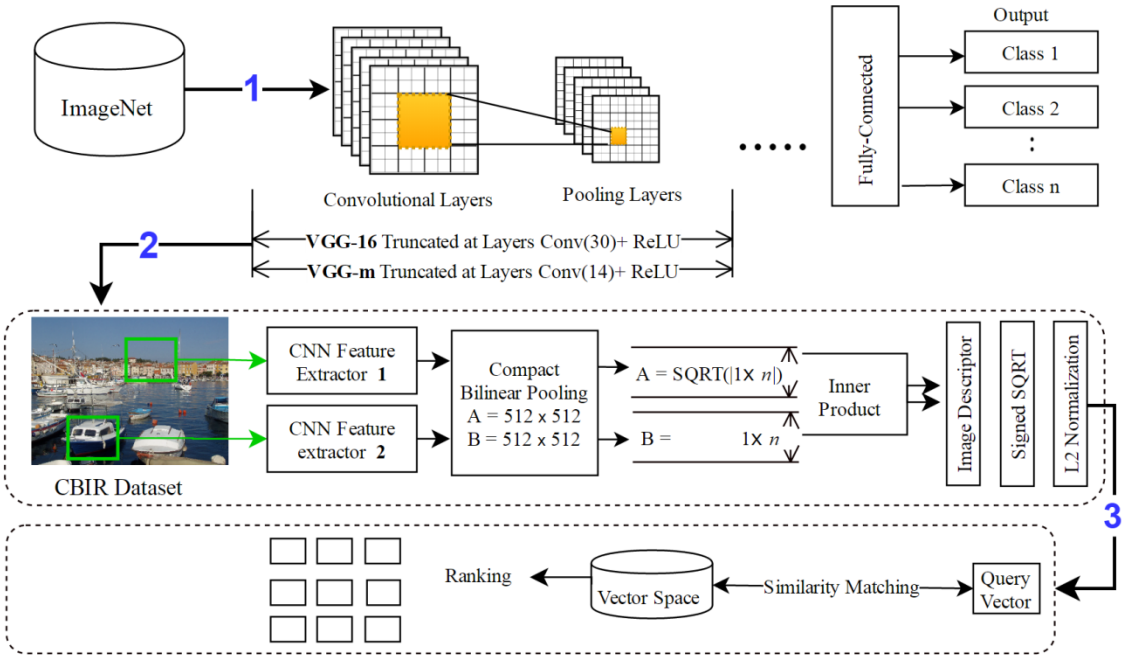


Fig.1. CRB-CNN architecture and retrieval framework

Firstly, the original CNNs are truncated at the last convolutional layer in each model where the output size is 512. Specifically, the first 14 layers are taken from VGG-m and the first 30 layers from VGG-16. Secondly, three additional layers are added to the end of the resulting CNN architecture as follows: 1) root compact bilinear pooling to project the data into small size of N; 2) SQRT; and 3) L2 normalization. The resulting network is then fine-tuned on domain-specific (image retrieval) datasets using an end-to-end training. The low-dimensional bilinear features extracted from this network are formed into a single generic image vector by the inner product between the outputs of two extractors. Finally, as shown in the lower part of Figure 1, the final architecture is used to extract features of queries and dataset images.

It is clear that all of images in our retrieval approach are indexed into one dataset of image vectors whose distance scores to each query image are computed. Several distance measures (e.g. Euclidean and Manhattan) are utilised to rank images then compute the retrieval accuracy and performance. The challenging task for the resulting network is the capability of preserving high discriminative image representations with a very compact size, which is a centric part of our architecture. An end-to-end training is conducted using unsupervised training, i.e. we group each dataset images into a set of classes according to their standard semantics but the network does not use any image labels, annotations or bounding boxes during the training and retrieval processes. Moreover, many critical issues in the context of image retrieval are considered; including the computation complexity, memory usage, and speed of feature extraction. Extensive experiments are carried out on the resulting architecture under several retrieval scenarios which is discussed in Section 4.

3.2. Feature Extraction and Retrieval

Given a pre-trained CNN network (VGG-m or VGG-16) with L layers, an input image I is warped into an 224x224 square to fit the size of training images then passed through the network in a forward pass of E epochs after applying the filters to the input image. In the last i-th convolutional layer, i.e. C_{14} and C_{30} in VGG-m and VGG-16 respectively, image features are accumulated at various locations and scales using the convolutional activations with a default output size of 512 for each of the two parallel CNNs. Specifically, a bilinear model B for CBIR task can be formulated a triple B = (f_A, f_B, P). Both f_A and f_B are feature functions, and P is a pooling function. Given an image I and a location L, a feature function is a mapping $f : L \times I \rightarrow R^{a \times D}$, where $R^{a \times D}$ is the feature output of size $a \times D$. Therefore, image features are combined by the bilinear feature combination of f_A and f_B at each image location l using the matrix outer product as follows:

$$Bilinear(l, I, f_A, f_B) = f_A(l,I)^T \cdot f_B(l,I) \quad (1)$$

Hence, to construct an image vector the pooling function P aggregates the extracted bilinear features across all image locations. However, the bilinear pooling layer generates high dimensional features of size $a \times 512$ by each single CNN and accumulated by sum pooling, where a is the number of image local locations. Therefore, the dimension of pooled descriptors computed by the outer product between the resulting matrices is 512×512~= 262K. This high-dimensionality of image descriptor is unwieldy in the context of image retrieval where the indexing complexity and memory size are among of the most critical concerns. Therefore, a low-dimensional projection is applied to the extracted features using the root compact bilinear pooling.

The principal component analysis (PCA), which is one of the most commonly used approaches for dimensionality reduction, was initially applied to reduce the feature size at the testing level of image retrieval. The experimental results show that the retrieval accuracy is noticeably degraded after the PCA projection compared to the original size (262K). The reason is that because of the high dimensionality of resulting bilinear features (~262K) so that the PCA will be expensive and not suitable to get the principal components. Accordingly, another reduction approach is adopted in our pooling layer; specifically the compact pooling recommended in [19]. As a result, our architecture replaces the bilinear pooling layer by its compact version but with efficient modification. We apply the square root on only one of the two bilinear descriptors generated after the reduction and before performing the inner product to form the image vector as detailed in the following:

Given two sets of local features $X = \{x_1, \ldots, x_{|SP|}, x_{SP} \in R^a\}$ from image I_1 and $Y = \{y_1, \ldots, y_{|SP|}, y_{SP} \in R^a\}$ from image I_2. The features are extracted using the last convolutional layer of the CNN network, where SP is a set of spatial locations. An image vector can be formed into ($a \times a$) matrix using our modified root bilinear (RB) pooling as follows:

$$RB(X) = \sum_{sp \in SP} r(x_{sp}) x_{sp}^T \quad (2)$$

where $r(x_{SP}) = SQRT(|x_{SP}|)$. By considering the kernelized version of bilinear pooling to compare X and Y of two images using the second order polynomial kernel:

$$\langle RB(X), RB(Y) \rangle = \langle \sum_{sp \in SP} r(x_{sp}) x_{sp}^T, \sum_{sp \in SP} r(y_{sp}) y_{sp}^T \rangle$$
$$= \sum_{sp \in SP} \sum_{sp \in SP} \langle r(x_{sp}), y_{sp} \rangle^2 \quad (3)$$

Then, any low-dimensional projection function applied to approximate image features into $\phi(x) \in R^{dim}$ and $\phi(y) \in R^{dim}$, where $dim << c^2$, by calculating:

$$\langle RB(X), RB(Y) \rangle \equiv \langle RB(X)_{compact}, RB(Y)_{compact} \rangle$$
$$\approx \sum_{sp \in SP} \sum_{sp \in SP} \langle \phi(x_{sp}), \phi(y_{sp}) \rangle \quad (4)$$

In this modified version of compact pooling, the projection function random maclaurin (RM) [21] is applied as recommended in [19]. This rooted compact bilinear pooling has the advantages of breaking any symmetry property that may hold between the extracted features of the two CNN extractors; and largely increasing the retrieval accuracy due to the increment of descriptor's discrimination level, as presented in Section 4.2.

The resulting low-dimensional descriptor by the root compact pooling is then passed to the next layers, i.e. the SQRT and L2 normalization, as shown in Figure 1. The process of feature extraction, pooling, and retrieval is simplified and summarized in Algorithm 1. The final fine-tuned architecture by backpropagation is used to extract the image vectors for both queries and dataset images in order to compute the distance scores then rank the retrieved images. Extensive experiments are carried out and discussed in Section 4 under different scenarios of network modelling, feature extraction, and projections.

Algorithm 1. Feature Extraction and Retrieval Procedures

Input: query images $\boldsymbol{Q}$, dataset of $\boldsymbol{N}$ images, $\mathbf{C}_L$: last convolutional layer
Output: ***mAP*** of all queries **Q**

```
-Begin
    +For i = 1 to N do
        *ParFor cnn = 1 to 2 do                    // Two CNN feature extractor run in parallel
            1. image = resize [image(i), 24,24]    // Fit the size of input layer
            2. F = extractC_L(image)               // F: convolutional features of size: w×512
        *End
            3. F_dim = rootBilinearPooling(F1,F2)  // Generic image vector
            4. V = sign(V) .* (absolute(V))        // Signed square root
            5. V = V ./ norm(V+eps)                // L2 normalization
                                                   // eps: floating-point relative accuracy
    +End
        ~For q = 1 to Q do                         // For all query images Q
            6. V_q = repeat steps (1-6)
            7. scores = distanceMeasure(V_q , V_N)
            8. topRanked = sort(scores)
        ~ End
            9. mAP = computeMAP()                  // mean Average Precision
-End
```

Function: rootBilinearPooling

Input: F_1 from extractor CNN_1, F_2 from extractor CNN_2 , where both have have size of $dim \times 1$
Output: feature vector F_{dim} of size *dim*

```
-Begin
    1. [F1 , F2] = compactProjection(F1 , F2)   // New F of size: dim×1
    2. F1 = absolute(F1)
    3. F1 = sqrt(F1)
    4. F_dim = F1 .* F2                         // inner product
-End
```

4. EXPERIMENTS AND DISCUSSION

4.1. Image Dataset and Evaluation

Holidays dataset [22**]**. It is one of the standard benchmarking datasets commonly used in the CBIR to measure the robustness against image rotations, viewpoint and illumination changes, blurring, etc. The dataset consists of 1491 high resolution images with a large variety of scene types, e.g. natural, man-made, water and fire effects, etc., as shown in Figure 2 (top row). The dataset contains 500 image groups that represent distinct scenes. The first image of each image group is the query image and the correct relevant images are the other images of the group. For each query of 500 standard queries initiated in the retrieval system, the top relevant images are retrieved and ranked according to the similarity scores and then the standard *mAP* is computed to measure the retrieval accuracy based on the ground truth of Holidays dataset. The precision of ranked images is computed to measure the retrieval accuracy for any query as follows:

$$P(R_k) = \frac{\#(\text{relevant images} \cap \text{retrieved images})}{\#(\text{retrieved images})} \tag{5}$$

where the retrieved images are all of top images retrieved (R_k) and the relevant images are only images relevant to the query image according to its ground truth. For a single query image, the average precision (*AP*) is the average of the precision value obtained for the set of top *k* images existing after each relevant image is retrieved, and this value is then averaged over all queries in the image category. Therefore, if the set of relevant images for a query $q_j \in Q$ is $\{I_1, \ldots, I_m\}$ where Q is the set of all queries, then the man average precision (*m*AP) is defined as:

$$m\text{AP}(Q) = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{m}\sum_{k=1}^{m}P(R_k) \tag{6}$$

Oxford buildings dataset [23**].** The Oxford Buildings dataset consists of 5062 images collected by searching for particular Oxford landmarks, as shown in Figure 2 (middle row). The collection has a comprehensive ground truth for 11 different landmarks, each represented by 5 possible queries. This results in a set of 55 queries over which the retrieval system can be tested and evaluated using *mAP* as computed for Holidays dataset.

UK-bench dataset [24**].** It consists of 10200 images of 2250 different objects. Each object image is taken under four different viewpoints to get four visually similar images, as shown in Figure 2 (bottom row). The first image of each object category is taken as query so 2250 queries are initiated in the experiments. The standard accuracy measure used for the UK-bench dataset is computing the precision at top 4 images then the results averages over all queries. The best accuracy can be achieved is 4, e.g. 1 indicates only one relevant image to the query is retrieved at top 4 images, and 4 indicates all relevant images are successfully retrieved and ranked.

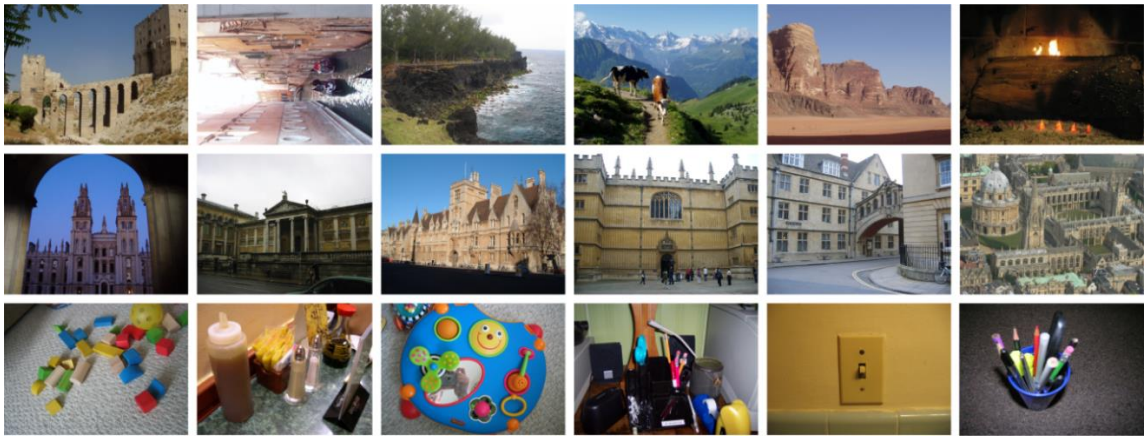


Fig.2. Samples of image datasets: Holidays (top row), Oxford (middle row), and UK-bench (bottom row).

4.2. Experiments Setup

Two scenarios are examined in all experiments using the proposed model and denoted as compact root bilinear CNN (CRB-CNN). For simplicity, we refer the two models as: CRB-CNN-(M) and CRB-CNN-(16) for the VGG-M network and VGG-verydeep-16 network, respectively. The specifications of each scenario are listed in Table 1. In all experiments, the last layer is first trained using the logistic regression followed by fine-tuning the whole resulting model on the retrieval dataset using back-propagation training for a number of epochs (between 30 and 80). A small learning rate is set which is then changing gradually, and two scales are chosen in all experiments because it shows the best performance among a range of scales. Once the trained model is generated, it is then used to extract the features of all queries and images in the benchmarking datasets, i.e. Holidays, Oxford, and UK-bench.

Table 1. Specifications of the two scenarios performed in all experiments.

Model	Layers	Truncated at	Initial length	Compact lengths	Distance Measure
CRB-CNN-(M)	19	$Conv_{14}$+ReLU	512×512	16,32,64,128,256,512	L1, L2, CityBlock
CRB-CNN-(16)	35	$Conv_{30}$+ReLU	512×512	16,32,64,128,256,512	L1, L2, CityBlock

Basically, the following procedure is adopted to rank and show the top images for every query image: Let C be a collection of N images; let v_n be the feature vector extracted from image n, where $n \in 1,...,N$; let $d(v_1, v_1)$ be a distance function defined between two vectors in the feature space; and let v_q be the feature vector corresponding to a given query image. Accordingly, the images n_{sim} in C most relevant and similar to the query image are the ones whose feature vectors minimize the distance to the query's feature vector:

$$n_{sim} = \arg\min_{1 \le n \le N} d(v_q, v_n) \qquad (7)$$

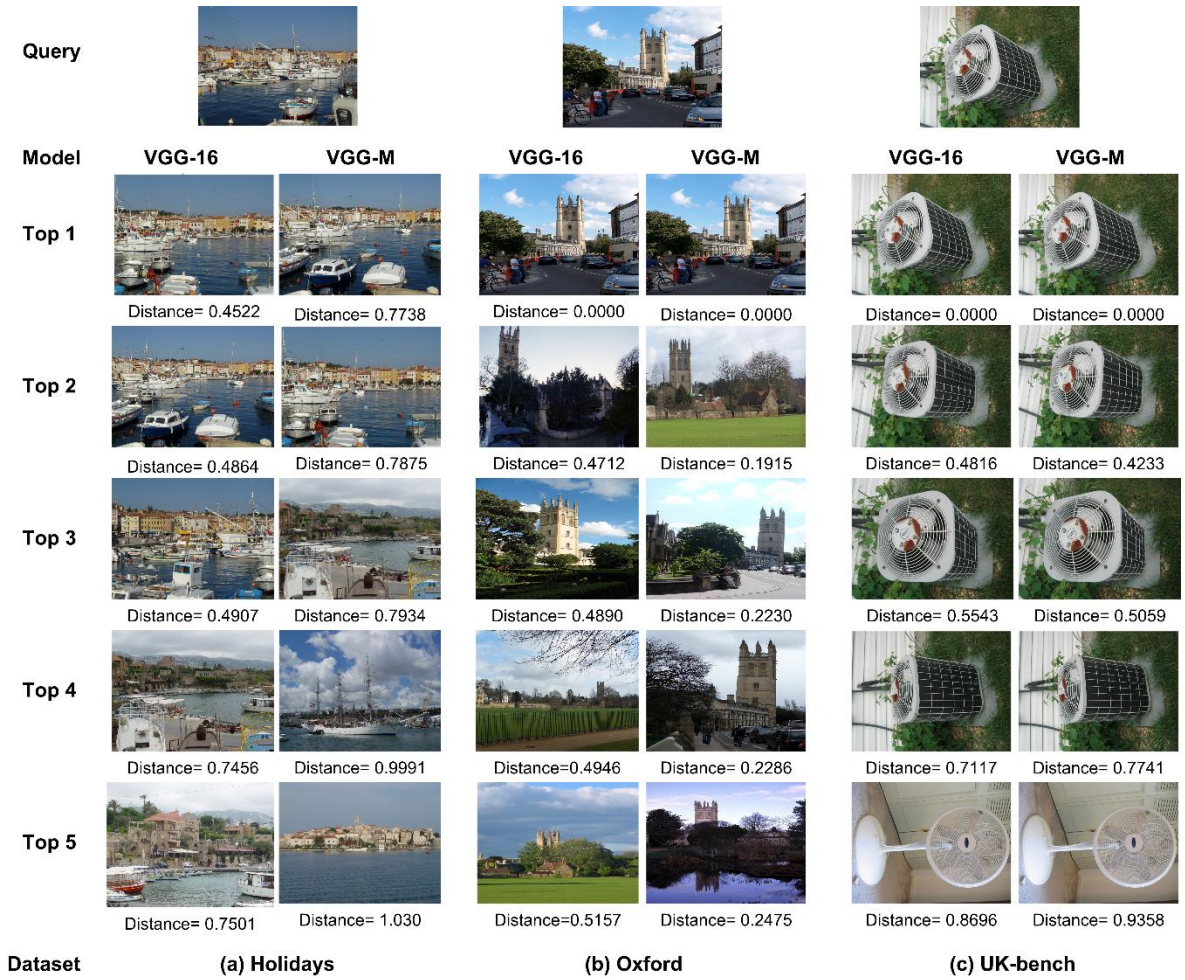


Fig.3. Sample queries and their top 5 similar images ranked from Holidays (a), Oxford (b), and UK-bench (c).

All image descriptors are then indexed into a single dataset structure, i.e. database (DB) of image vectors. To record distance scores, every query image's vector is compared against all of DB vectors by using three common distance measures: Manhattan (*L1*), Euclidean (*L2*), and Cityblock. Images are then ranked according to their obtained scores. Finally, the most commonly used measure (*m*AP) is used to evaluate the retrieval accuracy, i.e. *m*AP, and computed based on the evaluation protocol of image dataset, i.e. considering only the top-*k* relevant images to each query.

For each model, the dimensionality of image features is reduced using the root compact bilinear pooling to a range of compact dimensions: (512), (128), (64), (32) and (16). Figure 3 shows sample results of the two models used to retrieve and rank the relevant images to each query image (top row). A query vector of small size and Euclidean distance are used to show the top 5 images. The ranked images to all queries are similar and belong to the same semantic group according to each dataset ground truth. In addition, different distances and positions are given to the ranked images by the two models, i.e. CRB-CNN-16 and CRB-CNN-M.

4.3. The Retrieval Accuracy on CBIR Tasks

In this section, the retrieval performance is evaluated in terms of accuracy, speed, and memory usage. The two models CRB-CNN-(16) and VGG-(M) are evaluated on the three image datasets that represent different CBIR tasks, i.e. Holidays for general retrieval, Oxford for landmark retrieval, and UK-bench for object-focused retrieval. The results of retrieval accuracy (*m*AP) of Holidays, Oxford, and UK-bench are shown in Table 2, Table 3, and Table 4, respectively. Firstly, the accuracy results reported in Table 2 show that the very deep model CRB-CNN-(16) outperforms CRB-CNN-(M) by achieving better accuracy at most of vector lengths. Moreover, it is very clear that for both models the accuracy generally tends to increase when the vector length is decreased; emphasizing the astonishing capability of the proposed architectures on preserving high discrimination level while reducing the vector size.

Table 2. The retrieval accuracy *mAP* on Holidays dataset.

		Image Vector Length					
Model	**Distance**	**512**	**256**	**128**	**64**	**32**	**16**
CRB-CNN-(16)	Euclidean	85.44	89.30	93.20	95.13	86.54	81.85
	Manhattan	85.44	88.51	92.48	93.24	60.68	17.59
	CityBlock	85.20	88.60	92.92	94.71	85.66	80.44
CRB-CNN-(M)	Euclidean	80.95	81.16	84.10	86.34	85.55	89.04
	Manhattan	80.57	81.32	84.23	85.13	74.12	09.91
	CityBlock	80.97	81.53	84.42	86.02	84.62	87.62

Table 3. The retrieval accuracy *mAP* on Oxford dataset.

		Image Vector Length					
Model	**Distance**	**512**	**256**	**128**	**64**	**32**	**16**
CRB-CNN-(16)	Euclidean	69.73	71.01	82.62	84.14	91.69	95.74
	Manhattan	68.94	70.32	81.94	83.48	86.59	56.04
	CityBlock	68.98	70.43	82.01	84.03	90.55	94.72
CRB-CNN-(M)	Euclidean	58.15	61.06	64.30	72.14	81.82	85.67
	Manhattan	57.92	60.02	62.64	68.65	76.77	40.22
	CityBlock	57.84	60.05	62.84	71.07	80.73	84.67

Table 4. The precision at top 4 on UK-bench dataset.

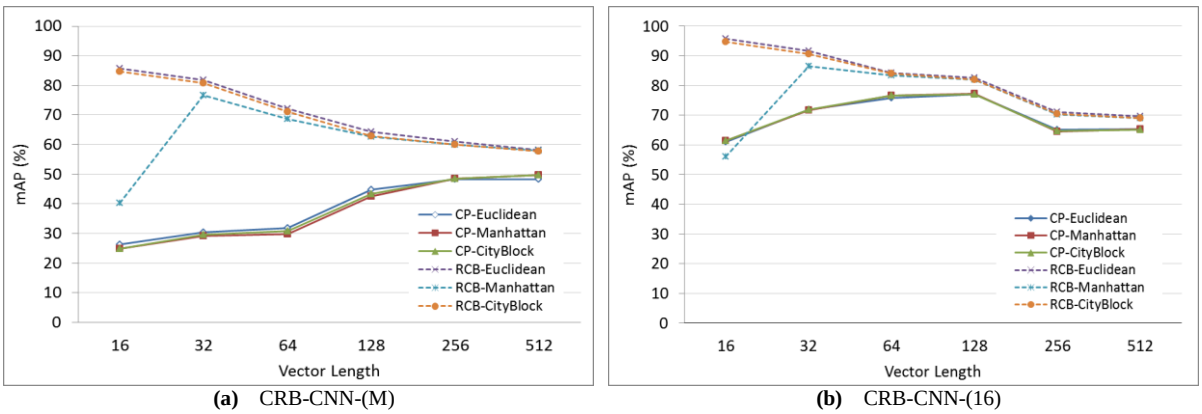
		Image Vector Length					
Model	**Distance**	**512**	**256**	**128**	**64**	**32**	**16**
CRB-CNN-(16)	Euclidean	3.56	3.49	3.33	3.09	2.46	1.81
	Manhattan	3.54	3.48	3.30	3.01	1.85	0.29
	CityBlock	3.54	3.47	3.30	3.03	2.40	1.78
CRB-CNN-(M)	Euclidean	3.40	3.35	3.19	2.95	2.46	1.78
	Manhattan	3.37	3.32	3.17	2.87	2.15	0.20
	CityBlock	3.37	3.33	3.17	2.89	2.41	1.76

Additionally, the distance measures used for similarity matching have shown different performance for each architecture on Holidays. Both models perform better using Euclidean distance; especially using vectors of size 128, 64, and 16. The best accuracy achieved by the CRB-CNN-(16) on Holidays dataset is 95.1% using image vector of size 64, while the CRB-CNN-(M) is performing better using image vector of size 16 with 89.04% accuracy. The results listed in Table 2 also show that the accuracy of both models is dramatically dropped using Manhattan distance on image vectors of size 32 and 16. For instance, the CRB-CNN-(16) performance is degraded by 32% and 75% when the image vector is reduced from 64 to 32 and 16, respectively.

Secondly, the accuracy results obtained by the two models on Oxford dataset and shown in Table 3 confirm the general trend of their performance. Specifically, they also perform better at the low image dimensions; especially at size 16 and the CRB-CNN-(16) and CRB-CNN-(M) achieve accuracy of 95.74% and 85.67%, respectively. These high accuracy values emphasize the efficiency of these models at retrieving very complex images even with many distractors exist in the most of images in Oxford dataset. Like in Holidays, the performance of both models is largely reduced at image when Manhattan distance is used at image vectors of size less than 64. As a result, high accuracy results obtained by the two models using the three distance measures except using Manhattan at vector of size 32 and 16.

Thirdly, Table 4 shows the retrieval results achieved on UK-bench image dataset. First of all, the training procedure applied on this dataset is different from what has been applied on the other image datasets. Specifically, we have just applied only one training pass, i.e. one epoch, in the forward-propagation and back-propagation for the CRB-CNN-(16) and CRB-CNN-(M). Despite this shallow binary-like training, the accuracy results reported in Table 4 are high using both models with the three distance measures. Unlike other datasets, the two models perform slightly better with image vectors of size between 128 and 512. These results affirm the effectiveness of the proposed architecture at learning visual features even with a large amount of image categories and variety of deformations, e.g. illumination, view-point, and rotation.

Finally, Figure 4 demonstrates the remarkable improvement in retrieval accuracy on Oxford dataset achieved by using the proposed root compact bilinear pooling (denoted as CRB with dotted lines in the figure) over the original compact pooling (denoted as CP with solid lines in the figure) and proposed in [19]. The results obtained in Figure 4(a) using the medium size CRB-CNN-(M) architecture show a noticeable increment in the retrieval accuracy on all vector lengths and distance measures, i.e. Euclidean, Manhattan, and CityBlock. Figure 4(b) also confirms the superiority of the RCB layer in improving the retrieval accuracy using the very deep CRB-CNN-(16) architecture. Roughly speaking, both CRB architectures with root bilinear pooling are largely performing better than the compact pooling; especially with the small vector lengths (16-128).



(a) CRB-CNN-(M) **(b)** CRB-CNN-(16)

Fig.4. The accuracy improvement achieved by our proposed root compact pooling over the compact pooling in [19].

4.4. Memory Usage and Search Time

As aforementioned, the compact size of image representations extracted by the CRB-CNN models is beneficial for the retrieval performance in terms of search time and memory size required to store the indexed images. In this section we provide more details on the retrieval performance the two models on all image datasets. All experiments in this work are reported using a CPU of 3.4GHz speed and 16GB RAM. The training process is carried out using the GPU NVIDIA Tesla K40.

Figure 5 (left part) shows the average time that the CRB-VGG-(16) takes to extract the image features and formulate them into a single vector in each dataset. It is clear that even with this very deep model the time ranges between 200 and 750 milliseconds (ms), which depend on the image visual contents and the dimensions at query submission. This average time is about the half (80 to 400ms) needed by the CRB-CNN-(M). However, this time is almost the same over all dimensions of image vector, i.e. 16 to 512. For example in UK-bench, the

model takes about 490ms to extract a vector of size 512 while takes 480ms for a vector of size 16. Therefore, it spends extra time on data approximation from size 262K to 16 than the time spent to reduce the same size to 512. As a result, the time of image extraction is nearly the same over all image dimensions using both the CRB-CNN-(16) and CRB-CNN-(M).

On the other hand, Figure 5 (right part) presents how largely the memory storage is reduced in order to store every single image in the dataset. It is clear that the memory size is dropping linearly when the vector size goes from 512 to 16. The storage size required for every single image ranges from only 1.75 KB to 0.06 KB using image vectors of 512 to 16, respectively. Roughly speaking, one million images will approximately require 1.67 GB of memory using image vector of size 512 and only about 58.6 MB required using image vector of size 16. This massive save on memory usage is very important in the context of image retrieval and directly affects the performance of the CBIR applications.

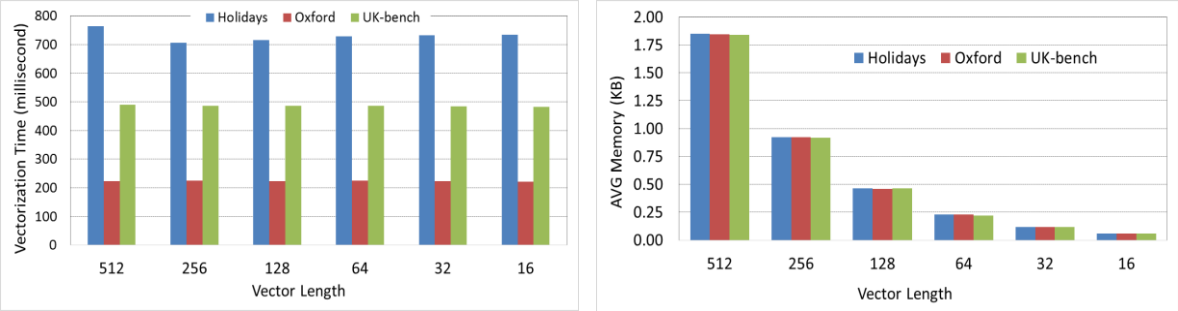


Fig.5. The vectorization time for each image in milliseconds (left), and the average memory size in Kilobytes to store a single image (right) using the CRB-CNN-16 model on a range of vector lengths.

Table 5. The average performance results on Oxford using VGG-16 and VGG-M models.

		Vector Length					
Oxford Dataset (5K images)		**512**	**256**	**128**	**64**	**32**	**16**
Average time of image vectorization (*millisecond*)	CRB-CNN-(16)	223	224	223	225	224	221
	CRB-CNN-(M)	84	82	86	81	85	85
Average time of query search (*millisecond*)	CRB-CNN-(16)	63	45	45	43	40	33
	CRB-CNN-(M)						
Average memory of single image (*KB*)	CRB-CNN-(16)	1.85	0.92	0.46	0.22	0.12	0.06
	CRB-CNN-(M)						

For instance, Table 5 provides more numerical details about the retrieval performance of our models CRB-CNN-(16) and CRB-CNN-(M) on Oxford dataset in terms of: 1) the average time required to extract every single image in milliseconds, 2) the average time needed to search the whole dataset including the similarity matching and image ranking, and 3) the average storage size required to store a single image on the disk. Foremost, both models (16 and M) have nearly the same average time required to search the whole dataset (5K images) and to sort the similarity scores for image ranking. This time takes 45ms in average over all vector dimensions, which is very fast due to the low-dimensionality of image vectors. In addition, the average time spent to extract the final image vector is 224ms in average for the very deep CRB-CNN-(16) and 84ms in average for the CRB-CNN-M model. Therefore, adding the time of image vectorization to the time of image search shows the efficiency of these models for the CBIR tasks. For example, the CRB-CNN-(M) takes only 129ms in average to extract the image vector and search the whole dataset with similarity matching using compact image lengths.

Finally, the average storage size required to store the indexed image is very small, e.g. less than 1KB for image of size <256 and only 61Bytes (0.06KB) for the image of size 16. As a result, the medium and very deep models proposed in this work show high performance in all critical steps that should be considered in any CBIR application; including features extraction, image search and ranking, memory usage, and most critically the retrieval accuracy itself.

4.5. Comparisons with the State-of-the-art

In this section, the accuracy results obtained by the CRB-CNN-(16) and CRN-CNN-(M) are compared on the three image datasets: Holidays, Oxford, and UK-bench. Table 6 shows the retrieval accuracy (*m*AP) using high-dimensional image vectors (upper part) and compact image vectors (lower part). It is clear that our models noticeably outperform the best results achieved by the local-based and CNN-based approaches even with high-dimensional image vectors. Additionally, the CRB-CNN-(16) and CRB-CNN-(M) models have superiority of using smaller image vectors than other approaches including the compact CNN-based methods. For Holidays, an improvement of 10% achieved over the best accuracy of compact vectors and 6% over the high-dimensional vectors. For Oxford, the accuracy is largely increased by 35% over other large and compact image vectors. Finally, our models with only one training epoch achieved high and comparable accuracy, 3.56, compared to the other methods on the UK-bench dataset; emphasizing the high ability of the CRB-CNN models at learning image features for different types of CBIR tasks, i.e. general, landmarks, and object image retrieval.

Table 6. Accuracy comparisons (*m*AP % and P@4) with the state-of-the-art.

Local and CNN-Based Approaches		**Holidays**	**Oxford**	**UK-bench**
Spatial Pooling [18]		89.7	84.4	-
Neural codes [26]		79.3	54.5	-
OxfordNet [13]		83.8	64.9	-
VLAD [3]		63.4	-	3.47
Triangulation-Embedding [27]		77.1	67.6	-
Label-image Embedding [28]		78.9	-	3.36
Improved BOW [29]		74.7	-	3.55
SERVE [30]		86.8	-	3.69
HVLAD [31]		72.1	63.8	3.56
Fine-residual VLAD [32]		62.2	-	3.43
S-sim [33]		84.0	78.7	3.24
Compact CNN-Based Approaches	**Length**	**Holidays**	**Oxford**	**UK-bench**
MOP-CNN [12]	512	78.4	-	-
NetVLAD-CNN [25]	256	86.0	63.5	-
Spatial Pooling [18]	256	74.2	53.3	-
Neural codes [26]	128	78.9	55.7	3.56
OxfordNet [13]	128	81.6	59.3	-
GoogLeNet [13]	128	83.6	55.8	-
CRB-CNN-16 (ours)	64	**95.1**	**84.1**	-
CRB-CNN-16 (ours)	16	81.85	**95.7**	-
CRB-CNN-M (ours)	64	**86.3**	**72.1**	-
CRB-CNN-M (ours)	16	**89.0**	**85.7**	-
CRB-CNN-16 (ours)	512	85.44	**69.73**	3.56

4.6. Large-Scale CBIR Using CRB-CNN Models

This section presents the retrieval accuracy (*m*AP) of the CRB-CNN model on Oxford5K-Flickr100K image dataset. Table 7 and Table 8 list the results obtained by the CRB-CNN-(M) and CRB-CNN-(16), respectively. Foremost, the retrieval accuracy achieved by both models on this large-scale dataset is high and comparable with the results reported on Oxford5K under the same setting and using the same set of queries and ground truth. This emphasizes the robustness of the CRB-CNN models in searching and retrieving within images with numerous variations in contents and semantics.

However, the accuracy results show that reducing the size of image vector eliminates most of unnecessary data while preserving the representative data with high discriminative level. This confirms the same conclusion reported on Oxford-5K, see Section 4.2. The best accuracy scores achieved by CRB-CNN-(M) is 73.30% using image representation of 16-dimension, while score of 88.63% is the best achieved by CRB-CNN-(16) also using 16-dimension. In consequence, the overall accuracy degradation compared to Oxford5K using an extreme dimensionality reduction (size-16) is about 11% using CRB-CNN-(M) and only 7% using CRB-CNN-(16). Again, the overall best scores of retrieval accuracy are obtained using Euclidean distance, followed by CityBlock then Manhattan, and the latter shows a noticeable degradation in the retrieval performance using images of size less than 32.

Table 7. The Retrieval Accuracy mAP(%) of CRB-CNN-(M) on Oxford105K.

Vector Size	Euclidean	Manhattan	CityBlock	DB Size on Memory (MB)
512	54.27	54.60	54.52	189.3
256	55.79	54.60	54.72	94.5
128	58.48	56.66	57.21	47.5
64	61.99	57.97	59.84	23.9
32	70.61	64.07	68.98	12.0
16	73.30	12.21	70.00	6.0

Table 8. The Retrieval Accuracy *m*AP(%) of CRB-CNN-(**16**) on Oxford105K.

Vector Size	Euclidean	Manhattan	CityBlock	DB Size on Memory (MB)
512	64.89	64.09	64.41	189.4
256	66.01	65.08	65.14	94.6
128	76.31	76.19	75.99	**47.0**
64	78.00	77.16	77.80	23.7
32	86.60	78.97	85.23	11.9
16	88.63	29.81	87.53	5.9

Scaling up the deep architectures challenges their effectiveness in transferring the learned visual semantics on a small amount of training images in order to retrieve the most relevant images to the initiated query. Firstly, the CRB-CNN performance is compared with the deep models proposed by Babenko et al. [26] on the full range of vector lengths, i.e. 16 to 512. Then, a comparison with other recent results reported on the large-scale image retrieval task is presented. Figure 6 shows the results obtained by: CRB-CNN-(M), CRB-CNN-(16), and neural codes approach [26]. It is clear that both models (M) and (16) using Euclidean measure outperforms the neural codes approach at all lengths of image representation; especially on the very compact ones, i.e. less than 64. In addition, the memory size required to store 105,000 images indexed by 16 dimensions is only about 6 MB, which is increasing linearly while the size of vector is increased.

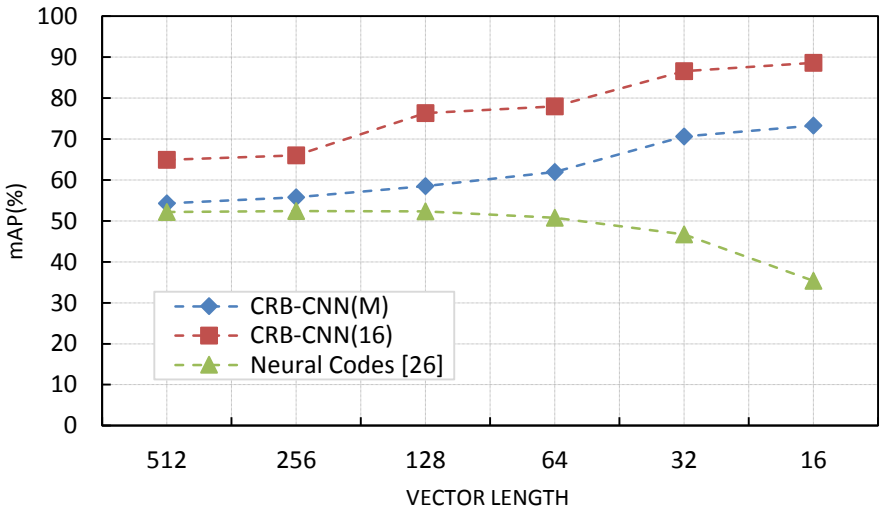


Fig.6. Accuracy comparisons with compact vectors on Oxford105K.

Finally, Table 9 list the best results recently reported by stat-of-the-art approaches on Oxford105K image dataset. The CRB-CNN outperforms the best recent scores of retrieval accuracy with minimum 9% and 24% improvement in accuracy using the CRB-CNN-(M) and CRB-CNN-(16), respectively.

Table 9. Comparisons with state-of-the-art on Oxford105K.

Vector Size	mAP %
Razavian et al. [17]	48.9
Babenko et al. [26]	52.4
Yandex and Lempitsky [14]	64.2
Jegou and Zisserman [27]	61.1
CRB-CNN-(M)	**73.3**
CRB-CNN-(16)	**88.6**

5. CONCLUSION

This paper introduces compact bilinear CNN-based architectures for several CBIR tasks using two parallel feature extractors without prior knowledge about the semantic meta-data of image contents. Image features are directly extracted from the activations of convolutional layers then largely reduced to very low-dimensional representations using the root bilinear compact pooling. The very deep architecture CRB-CNN-(16) and medium architecture CRB-CNN-(M) are fine-tuned for three CBIR tasks: general contents, landmarks, and object images. The extensive experiments conducted in this work provide several important conclusions. Firstly, the bilinear CRB-CNN models demonstrate high efficiency in learning even complex image contents belong to different semantic groups and include a lot of deformations and object distractors. Architectures initialization by deep models that are pre-trained on millions of images increases the level of image discrimination. Secondly, the CRB-CNN models reduce the image representations to very compact lengths, which remarkably boost the retrieval performance in terms of extraction and search time, and storage cost. Few tens of milliseconds (~ 130ms) are required to extract image features and search the database, and a small memory size (< 1KB) is needed to store the image on disk. Thirdly, the results obtained show that in most cases the extracted representation becomes highly discriminative when reducing the initial size (262K) of convolutional features to a range of compact vector lengths (512 to 16). Fourthly, Euclidean distance measure shows the best retrieval accuracy over all vector dimensions on the three image dataset, followed by the CityBlock measure with close accuracy results and then followed by the Manhattan measure which shows a noticeable degradation in the performance at vector lengths less than 64. Finally, an end-to-end training is applied without using any annotations, content tags, or other meta-data, which confirms the high capability of the CRB-CNN models in CBIR tasks using only the features extracted from only the visual contents. The developed deep models are also evaluated on large-scale image retrieval and showed high retrieval performance.

Acknowledgment. We gratefully acknowledge the NVIDIA for their generous donation of the GPU Tesla K40 used in this research.

REFERENCES

[1] G. Lowe, Distinctive image features from scale-invariant keypoints, International journal of computer vision 60, no. 2, 2004, pp. 91-110.

[2] H. Bay, T. Tuytelaars, L. Van Gool, SURF: Speeded up robust features, In Computer vision–ECCV, Springer, 2006, pp. 404-417.

[3] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, C. Schmid, Aggregating local image descriptors into compact codes, In IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(9), 2012, pp.1704-1716.

[4] A. Krizhevsky, I. Sutskever, G. Hinton, Imagenet classification with deep convolutional neural networks, In Advances in Neural Information Processing Systems, vol. 25, 2012, pp. 1106–1114.

[5] S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, In CVPR Workshops, 2014, pp. 806–813.

[6] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR, abs/1409.1556, 2014.

[7] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, arXiv preprint arXiv:1311.2524, 2013.

[8] M. Oquab, L. Bottou, I. Laptev, J. Sivic, et al., Learning and transferring midlevel image representations using convolutional neural networks, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1717-1724.

[9] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L. Fei-Fei, ImageNet: A large-scale hierarchical image database, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255.

[10] K. Lin, H.F. Yang, J.H. Hsiao, C.S. Chen, Deep learning of binary hash codes for fast image retrieval, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 27-35.

[11] J. Wan, D. Wang, S.C.H. Hoi, P. Wu, J. Zhu, Y. Zhang, J. Li, Deep learning for content-based image retrieval: A comprehensive study, In Proceedings of the ACM International Conference on Multimedia, 2014, pp. 157-166.

[12] Y. Gong, L. Wang, R. Guo, S. Lazebnik, Multi-scale orderless pooling of deep convolutional activation features, In Computer Vision–ECCV, Springer, 2014, pp. 392-407.

[13] J. Ng, F. Yang, L. Davis, Exploiting local features from deep networks for image retrieval, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 53-61 (2015).
[14] B. Yandex, V. Lempitsky, Aggregating Local Deep Features for Image Retrieval, In IEEE International Conference on Computer Vision, 2015, pp. 1269-1277.
[15] J. B. Tenenbaum, W. T Freeman, Separating style and content with bilinear models, Neural computation, 12(6), 2000, pp.1247–1283.
[16] T.Y. Lin, A. RoyChowdhury, S. Maji, Bilinear CNN models for fine-grained visual recognition, In Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1449-1457.
[17] S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, Visual instance retrieval with deep convolutional networks, In arXiv:1412.6574v3, 2015.
[18] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, C. Schmid, Local convolutional features with unsupervised training for image retrieval, In Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 91-99.
[19] Y. Gao, O. Beijbom, N. Zhang, T. Darrell, Compact Bilinear Pooling, In Proceedings of the IEEE International Conference on Computer Vision, 2016, in press.
[20] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: Delving deep into convolutional nets, arXiv preprint arXiv:1405.3531, 2014.
[21] P. Kar, H. Karnick, Random feature maps for dot product kernels, In International Conference on Artificial Intelligence and Statistics, 2012, pp. 583–591.
[22] H. Jégou, M. Douze, C. Schmid, Hamming Embedding and Weak geometry consistency for large scale image search, In the Proceedings of the 10th European Conference on Computer vision, 2008, 27.
[23] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1-8.
[24] D. Nistér, H. Stewénius, Scalable recognition with a vocabulary tree, In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, 2006, pp. 2161-2168.
[25] R. Arandjelović, P.Gronat, A. Torii, T. Pajdla, J. Sivic, NetVLAD: CNN architecture for weakly supervised place recognition, In Proceedings of IEEE International Conference on Computer Vision, 2016, in press.
[26] Babenko, A. Slesarev, A. Chigorin, V. Lempitsky, Neural codes for image retrieval, In ECCV, 2014, pp. 584–599.
[27] H. J´egou and A. Zisserman, Triangulation embedding and democratic aggregation for image search, In IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3310–3317.
[28] Gordoa, J. A. Rodríguez-Serrano, F.Perronnin, E. Valveny, Leveraging category-level labels for instance-level image retrieval, In Computer IEEE Conference on Vision and Pattern Recognition, 2012, pp. 3045-3052.
[29] H. Jégou, M.Douze, C. Schmid, Improving bag-of-features for large scale image search, International Journal of Computer Vision, 87(3), 2010, pp. 316-336.
[30] J. Li, C. Xu, M.Gong, J. Xing, W. Yang, C. Sun, SERVE: Soft and Equalized Residual VEctors for image retrieval, Neurocomputing, 2016, In press.
[31] Z. Liu, H. Li, W. Zhou, T. Rui, Q. Tian, Uniforming residual vector distribution for distinctive image representation, IEEE Trans. Circuits Syst. Video Technol. 99, 2015, 1-1.
[32] Z. Liu, S. Wang, Q. Tian, Fine-residual VLAD for image retrieval, Neurocomputing, 173, 2016, pp. 1183-1191.
[33] Y. Gao, M. Shi, D. Tao, C. Xu, Database saliency for fast image retrieval, IEEE Transactions on Multimedia. 17 (3), 2015, pp. 359–369.